Université Ibn Khaldoun – Tiaret
Faculté des Mathématiques et Informatique
Département d'Informatique

Matière : ASDD1
Niveau : 1 LMD (S1, S2, S3 et S4)
Année Universitaire : 2021-2022

Solution Examen Final

Le 18 /01/2022 Durée : 01^{H}:00^{mn}

Exercice 1 (*6 pts*). Selon la loi de finance 2022, l'Impôt sur le Revenu Global (IRG) d'un salarié est calculé d'une manière progressive, sur la base de son Salaire Imposable (SI) et conformément aux règles décrites dans le tableau ci-dessous.

Etant donné le SI d'un salarié, écrire un algorithme/programme C qui calcule et affiche son IRG.

Salaire imposable en DA (SI)	**Taux d'imposition**
SI < 20 000 DA	0%
20 000 DA < SI < 40 000 DA	23%
40 000 DA < SI < 80 000 DA	27%
80 000 DA < SI < 160 000 DA	30%
160 000 DA < SI < 320 000 DA	33%
SI > 320 000 DA	35%

Exemple.

- SI=36 000 DA, **IRG**=0*(20 000-0)+ 0.23*(36 000-20 000)=0+0.23*16 000=3680 DA
- SI=189 000 DA, **IRG**=0*(20 000-0)+ 0.23*(40 000-20 000) +0.27*(80 000-40 000)+ 0.30*(160 000-80 000)+ 0.33*(189 000-160 000)=0+4600+10800+24 000+0.33*29 000= =48 970 DA.

Solution Exo 1 :

```
int main(void) {
	int SI, IRG;
	printf("Donner le Salaire Imposable SI");
	do {
		scanf("%d", &SI);
	} while (SI < 0);
	if (SI > 320000)
		IRG = ((SI - 320000) * 0.35) + (160000 * 0.33) + (80000 * 0.3)
		        + (40000 * 0.27) + (20000 * 0.23);
	else if (SI > 160000)
		IRG = ((SI - 160000) * 0.33) + (80000 * 0.3) + (40000 * 0.27)
		        + (20000 * 0.23);
	else if (SI > 80000)
		IRG = ((SI - 80000) * 0.3) + (40000 * 0.27) + (20000 * 0.23);
	else if (SI > 40000)
		IRG = ((SI - 40000) * 0.27) + (20000 * 0.23);
	else if (SI > 20000)
		IRG = ((SI - 20000) * 0.23);
	else
		IRG = 0;
	printf("%d",IRG);
	return 0;
}
```

Exercice 2 (*7 pts*). On remplace la valeur d'un entier N donné strictement positif par la valeur N/2 (division entière) si N est pair ou bien par la valeur 3*N+1 si N est impair. On répète ce processus de changement de valeurs de N jusqu'à ce que la valeur de N devienne égale à 1.
Ecrire un algorithme/programme C qui Afficher la liste des valeurs obtenues par N et le nombre de fois qu'il a été nécessaire de changer N pour arriver à 1.

Exemple. Entier saisi, N=12

N=12 pair 12/2 = 6
N=6 pair 6/2 = 3
N=3 impair 3* 3+1 = 10

La liste des valeurs obtenues par N est :
6 3 10 5 16 8 4 2 1
Le nombre de fois qu'il a été nécessaire de changer N pour arriver à 1 est 9.

N=10 pair 10/2 = 5
N=5 impair 5*3+1 = 16
N=16 pair 16/2 = 8
N=8 pair 8/2 = 4
N=4 pair 4/2 = 2
N=2 pair 2/2 = 1 (Arrêt)

Solution Exercice 2 :

```
int main(void) {
	int n, nbFois;
	do {
		scanf("%d", &n);
	} while (n < 0);
	nbFois = 0;
	while (n > 1) {
		if (n % 2 == 0) {
			n = n / 2;
			nbFois++;
			printf("%d ", n);
		} else {
			n = 3 * n + 1;
			nbFois++;
			printf("%d ", n);
		}
	}
	printf("\nLe nombre de fois qu'il à été nécessaire de changer N pour arriver
à 1 est %d",
			nbFois);
	return 0;}
```

Exercice 3 (*7 pts*). Ecrire un algorithme /programme C qui permet de :

- Saisir N éléments dans un vecteur V (N>0).
- Permuter les éléments successifs du vecteur V (deux par deux)
- Afficher le vecteur V après permutation.

Exemple.

1	2	3	4	5	6	…..

Après permutation :

2	1	4	3	6	5	….

Solution Exercice 3

```
int main(void) {
	int N, i, c;
	do {
		scanf("%d", &N);
	} while (N < 0);

	int V[N];
	for (i = 0; i < N; i++)
		scanf("%d", &V[i]);
	printf("\nVecteur v:\n");
	for (i = 0; i < N; i++)
		printf("%d ", V[i]);
	for (i = 1; i < N; i+=2) {
		c = V[i];
		V[i] = V[i-1];
		V[i-1] = c;
	}
	printf("\nVecteur v après Permutation:\n");
		for (i = 0; i < N; i++)
			printf("%d ", V[i]);
	return 0;
}
```

Bon Courage